Deep learning model for predicting public transit arrival times

Armin Khayyer, Daniel F. Silva, and Alexander Vinel
Department of Industrial Engineering
Auburn University
Auburn, AL
Word Count: 5462 words +3 table (250 words per table) = 6212 words

January 9, 2020

Abstract

Predicting arrival times accurately in public transit can be difficult due to their inherent variability. We propose a Deep Learning tool to address this problem. The methodology combines an Artificial Neural Network approach that predicts based on static features and a Recurrent Neural Network which captures time series patterns. We use a New Orleans Transit Authority (NORTA) dataset collected by Automatic Passenger Counter (APC) devices to calibrate and test our method of forecasting for both buses and streetcars. The results of our experiments show a modest improvement compared to existing approaches for both modes.

***K*eywords** Bus Travel Time Prediction, Time Series, Neural Networks, Long Short-Term Memory Networks

1 INTRODUCTION

Traffic congestion in the U.S. has been increasing in recent years, taking a toll on the economy of estimated $300 billion per year [1]. Public transit systems are one way to alleviate this problem and have often been identified by researchers as a key component of the design and operation of so-called Smart Cities [2]. However, for several reasons the use of public transit in the U.S. has historically lagged behind the rest of the world [3]. One way to attract users of public transit, which has been deployed by many transit systems in the U.S., is to offer predictions of travel times and wait times. However, this may be counterproductive, since if the information provided is not reliable or accurate, customers in fact may be more likely to reject the system than if no predictive information were provided at all [4]. This places an emphasis on research and development of careful implementation methodologies of such forecasting systems.

The need to meet user expectations of travel time predictions, the ubiquity of massive data on trip details, and the proliferation of mobile technology have led to travel time prediction models and tools gaining a lot of attention in the literature in recent years. Several approaches have been proposed including techniques from classical statistics (i.e. regression models, time series models), Bayesian statistics (i.e. Kalman Filtering or KFT), and, more recently, machine learning.

Machine learning research on travel time forecasting initially focused on techniques such as Artificial Neural Networks (ANN) that predict travel time based on a feature vector, corresponding to each point in a large data set, with features such as the link distance, demand of passengers, etc [5, 6]. However, public transportation data is often highly auto-correlated. With recent advances in Recurrent Neural Networks (RNN), more and more RNN-based algorithms have been utilized in traffic prediction due to their ability to capture time series patterns and complex relationships from a huge amount of data [7, 8]. These models always take a sequence of travel time as input and try to predict the travel time for the following link, but do not use static features such as link distance, expected traffic or demand.

In this work, we propose a Deep Learning approach for predicting travel times of vehicles in a public transit setting. Our methodology learns spatio-temporal correlations by combining feature-based ANN model with an auto-regressive RNN model. The key idea is to feed the feature vector into an ANN model and the lagged travel time sequence into an

RNN model, then combine their outputs to predict the travel time for the following link. This architecture enables the model to learn the sequential long-term traffic pattern as well as the contributions of independent variables such as link distance, passengers demands, and so on. Although the dynamics of different modes of transit can be distinct from all other modes, our proposed model does not use mode-specific features and learns the traffic patterns, hence it is agnostic of the specific mode of transit.

We use data from New Orleans Rapid Transit Authority to calibrate and test our models using data from both buses and streetcars. We show that our approach produces modest improvements in both cases over using either a pure ANN or RNN approach. We also compare our approach to classical statistics techniques and conclude that either ANN, RNN or our approach all handily outperform straightforward linear regression and time-series methodologies.

The remainder of this paper is organized as follows, in Section 2, we provide a brief review of relevant literature on data driven approaches for travel time predictions, as well as some related efforts in which similar deep learning methodologies are utilized. Section 3 explains our methodology in detail. Section 4 describes the data sets we used and the experiments we carried out. Section 5 presents the results of our experiments, along with analysis and discussion. Finally, Section 6 summarizes our findings and conclusions.

2 RELATED WORK

The problem of accurately predicting travel times between stations for buses (or other modes of public transit) is challenging and hence has spawned several families of approaches for tackling it. Several factors, such as link distance, passenger demand at each link, traffic, etc. affect bus arrival times. These factors are then used as independent variables in predictive models. Many researchers have implemented statistical models, namely time series analysis and regression models, to forecast travel time. Assuming that the historical traffic patterns will remain the same in the future, time series models attempt to find a function that maps historical data to current traffic pattern. For instance, Al-Deek, Pizzo, and Wan (1998) [9] proposed a non-linear time series approach using only speed data. They showed that using models with one single variable can result in a reasonable errors for short-term travel time predictions. Alternatively, linear regression models attempt to predict and explain a response variable by fitting a linear equation to a set of independent variables. For example, Patnaik, Chien, and Bladikas (2004) [10] used Automatic Passenger Counter (APC) data in a multi-variable regression model to predict arrival time based on the numbers of passengers that aboard or alight from buses, distances from the station, dwelling time, and the station numbers. However, it must be noted that since these variables are often highly correlated, the application of linear regression might be challenging.

Kalman Filters (KFT) models work in two steps, it first attempts to provide estimates of the current state of the system based on a specified transition rule and the previous state. It then updates the estimated values using a weighted average of the estimation and the measurement, once the outcome of the next measurement is observed. Kalman Filters are especially popular for noise filtering (see, for example [11, 5, 12]) but have also been used for predicting bus arrival times. Vanajakshi, Subramanian, and Sivanandan (2009) [12] adopted a KFT Model to predict bus travel times and update the predictions as new observations become available, showing that their KFT Model outperforms a simple averaging method.

With the progress in machine learning, Artificial Neural Network models have also become popular for approximation and prediction purposes. In our specific application of predicting bus travel times, Kumar et al (2014) [13] showed that if there is a sufficiently large amount of data, then ANN models can outperform the KFT models. This might be because of the limited ability of KFT models in capturing non-linear long-distance patterns in traffic. Fan and Gurmu (2015) [14] take a different approach and develop regression, KFT and ANN models that only use GPS data as inputs (ignoring both features and historical travel times). They concluded that their ANN model performs better than their own KFT and regression approaches, but they did not compare their results to other methods.

Long short-term memory (LSTM) neural networks [15] are a Recurrent Neural Network architecture [16] that has been successfully applied in many real-world problems involving sequential data, such as image captioning [17], speech recognition [18], and neural machine translation [19]. It also has been used in bus travel time prediction. Duan, Lv, and Wang (2016) [7] showed that LSTM model is promising in traffic series data forecasting. Ran *et al.* [8] also proposed an LSTM based method with an attention mechanism. The attention mechanism is designed to process the output layer of each LSTM unit, learning accumulation of features patterns. They showed that their model can achieve a better accuracy than regular LSTM models.

So far, to the best of our knowledge, there have not been any attempts made at combining feature-based ANNs with time-series approaches for predicting travel times in public transit. This is a strategy that has been used successfully for forecasting other aspects of public transit, namely passenger demand. Ke *et al.* [20] proposed a so called "fusion" convolutional long short-term memory network for the short-term passenger demand forecasting, which consists of

multiple convolutional long short-term memory (conv-LSTM) layers [21], standard LSTM layers, and convolutional layers [22]. In their architecture spatio-temporary variables (sequences of travel time rate and sequences of demand intensity) are fed to conv-LSTM, while non-spatial time-series variables, namely time-of-day, day-of-week, and so on, are fed to LSTM layers. Then a weighted combination of all the layers is used to calculate the Mean Square Error loss. They showed that their proposed model outperforms the baseline approaches, time-series prediction models, and ANN based models. Furthermore, they showed that the consideration of variables such as the travel time rate, time-of-day, day-of-week, and weather conditions, is promising. We use a rather similar strategy, combining several machine learning approaches to improve travel time predictions.

3 METHODS

We use a combination of ANN and RNN models to generate travel time predictions. Specifically, we use a multi-layer perceptron (MLP) model to make predictions based on features and a long short-term memory (LSTM) neural network to incorporate auto-correlated time series data. Below we describe each model separately and then discuss how they can be combined leveraging the strengths of each.

MLP is a classical class of feed-forward ANN, consisting of at least three layers of nodes: an input layer, a hidden layer and an output layer. A non-linear *activation function* such as rectified linear, sigmoid, tanh and so on, is used for each node in the hidden layer as well as the nodes in the output layer. The back-propagation technique is used for training the MLP Network. Changing the connection weights based on the amount of error in the output compared to the labeled result is referred to as the learning process. We utilize this supervised learning method to approximate the mapping function between the feature vector (the set of independent variables) and travel time.

A MLP network has no notion of the order of training instances, and the only input it considers is the current observed data point. RNNs, on the other hand, take as input the sequential information that has been accumulated. The hidden state (h_t) of the RNNs preserve this sequential information. The output of a non-linear function of the weighted average of the input of the current time step and the hidden state from the previous time step is preserved as the hidden state value of that time step. Since at each time step the hidden state from the previous time step is used for both predicting an output and for updating the state, this state contains traces of the previous hidden states. This property of RNNs makes them a good fit for time series prediction. Training standard RNNs, however, can be challenging when solving problems with long term temporal dependencies, since the gradient of the loss function decays exponentially with time, which is often referred to as the vanishing gradients [23].

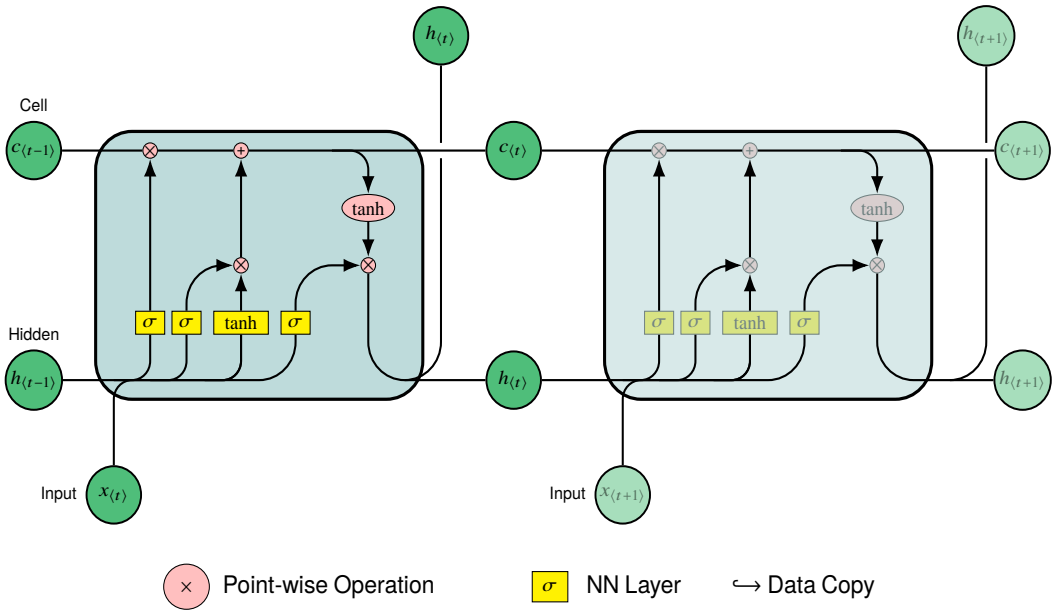


Figure 1: A diagram of an LSTM network

A common LSTM network [15] is a special design of RNN which is composed of LSTM cells, each consisting of an input gate, an output gate, and a forget gate (Figure 1). The three gates are used to control information flow (cell state), determining what extent of the current data is used to update the cell state, what extent of the cell state value to forget,

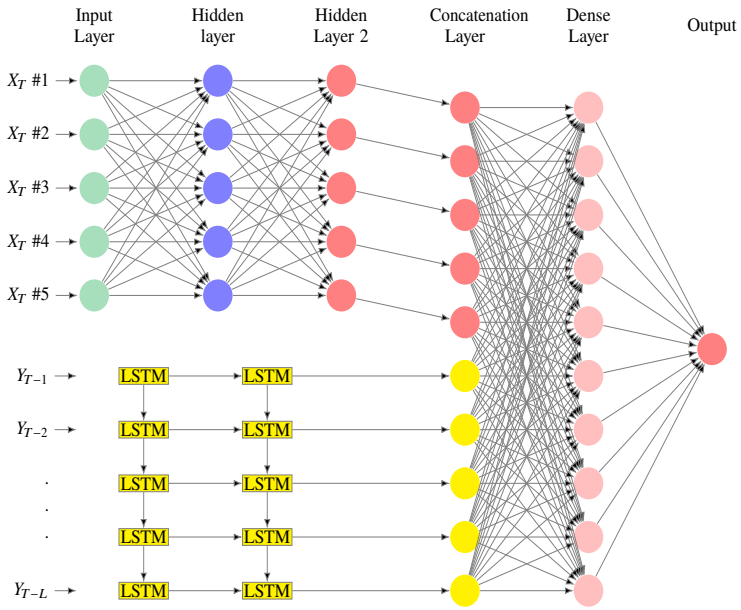


Figure 2: Combined MLP-LSTM model architecture

and to output. The architecture is designed to promote learning of longer-term dependencies. LSTM networks are well-designed for classifying, processing and making predictions based on time series data, since there can be lags of unknown duration between important events in a time series. We use this property of LSTM's to take into account the auto-correlation in our dataset. Moreover, if the traffic pattern changes over time, the forgetting property of LSTMs enables the model to dismiss the previously learned data, thus allowing for adapting to new conditions.

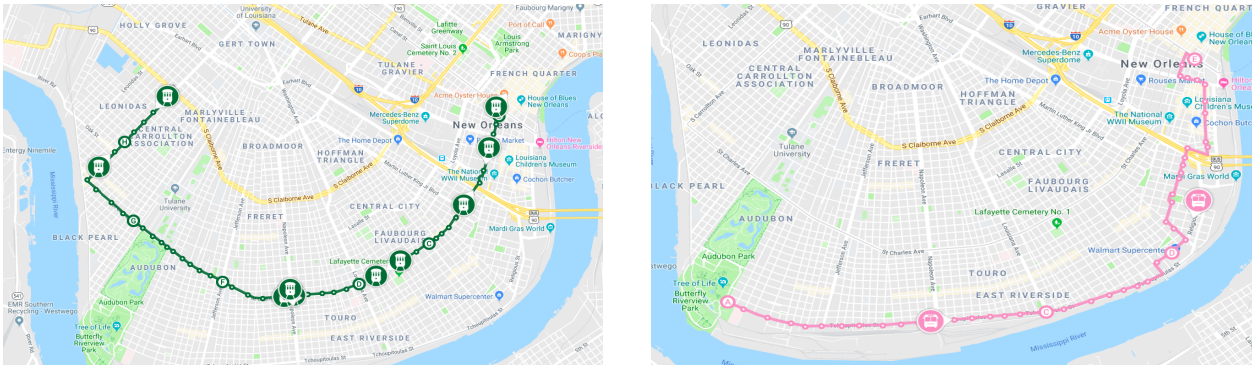
In the proposed model, we utilize an LSTM network to capture time series patterns and a MLP model to learn from independent variables. The architecture of the model is shown in Figure 2. A constructed lagged travel time vector, which is a vector of previous links' travel times, is fed to an LSTM model. The number of previous links to look back is a hyper-parameter of the model. Simultaneously, the vector of independent variables for each data point is fed to an MLP model. We then concatenate the outputs of both models and feed them to a fully connected layer. Finally, we calculate the loss function, Mean Squared Error (MSE) between the ground truth label and the predicted travel time, and then train the model with back-propagation. The idea is to take into consideration both the auto-correlation and the relationship among the travel time and the explanatory variables. The last layer, which is fully connected, learns the weights associated to the output of each section during the training process.

4 EXPERIMENTS

We used a historical dataset of several buses and streetcars, equipped with APC devices, operating on different routes in New Orleans provided by the operator of the New Orleans Regional Transit Authority (NORTA). APCs were used to collect location measurements, longitude and latitude, passengers count, and matched stops. Each row of the dataset was collected consecutively along the route with each row representing a stop. The dataset contains several columns, namely date and time, number of passengers boarding and alighting the bus at each step, direction, bus load, the longitude and latitude of the bus stop, and so on.

To test our model, we chose two routes from the dataset, a bus route, Route 10 – Tchoupitoulas, and a streetcar route, St. Charles Streetcar. This allows us to compare the model's predictive power across different modes of public transit. Figure 3 shows a map of each of the routes. We specifically chose routes that have high demand, and operate in parts of the city with highly variable traffic. Such types of routes are naturally more difficult to predict as the data is noisier. Furthermore, the two selected routes have high frequency, implying that users are more likely not to rely as much on schedules [24], but instead are interested in accurate forecasts of waiting times.

As mentioned before, data points were collected consecutively while the buses were operating on a specific route. By subtracting arrival time at each stop from its predecessor, we compute the time it takes to travel from one stop to the next. We consider the computed travel time as the ground truth labels. Assuming that the traffic pattern is roughly the same during different seasons, we collected traffic measures (Estimated Travel Time) for each link using Google Maps



(a) St. Charles Streetcar (b) Route 10 – Tchoupitoulas

Figure 3: Selected streetcar and bus routes

APIs for several weeks. Then we averaged the collected data for each specific hour of weekdays, which we then use as an independent variable for travel time prediction. Figure 4 shows the time series traffic patterns collected. It also shows peak hours during mornings and afternoons. We used these measures instead of categorical data for hour of the day, since it contains more information. For each data point we assign the corresponding traffic measure based on its location, hour of the day, and day of the week. We observed that this measure has a significant contribution to travel time prediction.

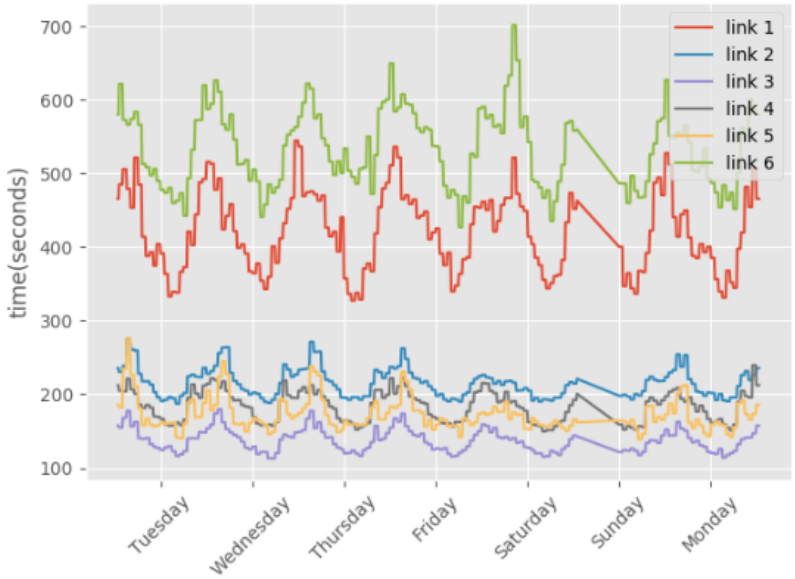


Figure 4: Estimated travel time collected from Google APIs. St. Charles Streetcar route is partitioned to links 1 through 6. The y-axis shows the travel time in seconds for each link, while the x-axis shows hours of the day for an entire week.

It is worth noting that the proposed model is robust with respect to missing data points, which is a significant advantage. Missing data can occur for two reasons. First, APC devices may not accurately match stops to the collected data points, since GPS signals can be lost sporadically, for example when buses pass by tall buildings [25]. In addition, by visualizing the collected GPS points, we observed that some had been collected either at the bus depot or when the bus was not operating on the specified route. Removing both of such irrelevant non-en-route or incorrectly GPS-labeled points results in missing data, yet by construction, the model is still applicable. Specifically, since the distance and time are calculated as the differences between two consecutive points, we do not need every data point to be present. Furthermore, since the MLP model only considers the set of independent variables for encoding, it has no notion of time and hence is agnostic to missing lines. Finally, the LSTM, because of its long short-term memory, can remember information from several previously observed data points of the bus operating on the route.

As a pre-processing step, we normalized the dataset using the Min-Max Scaler from scikit-learn [26], since LSTMs and MLPs are sensitive to the scale of the input data, specifically when the sigmoid or tanh activation functions are used. The pre-processed streetcar and bus datasets have 44,000 and 37,000 data points respectively. We used 80% of each for training the model and the remaining 20% for testing it.

We used multiple linear regression as a benchmark to compare our machine learning approaches. Multiple linear regression attempts to model the relationship between two or more explanatory variables and a response variable by fitting a linear equation to the observed data. Every value of the independent variables x_i is associated with a value of the dependent variable y. In our model, the computed travel time is the dependent variable. We used a feature selection algorithm over our dataset to find the most significant variables for prediction. The initial variable set included: bus load, distance, number of passengers boarding, number of passengers alighting, traffic measure, day of the week, categorical peak time, and direction (inbound or outbound). Step-wise regression resulted in the following variable set: bus load, distance, number of passengers boarding, number of passengers alighting, and the traffic measure collected from Google APIs, all of which are statistically significant.

We used a three-hidden-layer feed-forward neural network as a baseline model to predict the travel time. For this model we used the same input variables selected based on the step-wise regression algorithm. We used TensorFlow [27] for designing the MLP model. We used three hidden layers, each with 10 nodes. This model architecture was arrived at by empirically fine tuning the model over our dataset. We used Adam optimizer [28] with 0.01 learning rate for 200 epochs. We also used a rectified linear (relu) function as the activation function, and Mean Square Error (MSE) for the loss function.

We used Keras [29] sequential method to build our proposed model. For training the model a vector of lagged travel times as well as a feature vector are needed for each data point. The MLP section of the model has the same structure as the baseline MLP model. We chose empirically to look back over the last 30 travel times at each stop. We feed the lagged vector into 30 units of LSTMs (Figure 1), then we concatenate the LSTM outputs with the encoded feature vector of each row, computed by the MLP section of the model. Next, we feed forward the concatenated matrix into a dense fully connected layer with 5 nodes and *relu* activation function. Finally, we feed forward the fully connected layer into one single output neuron, and compute the MSE Loss. We used Adam optimizer [28] with 0.01 learning rate to train the model.

We compare our proposed model with four simpler approaches namely Linear Regression (referred to as LR), Linear Regression with lagged variables (LR-lagged), our baseline MLP model, and the MLP model with lagged variables (MLP-lagged). A lagged model in statistics is defined as a time series model which tries to predict current value of a dependent variable based on the previously observed values of the explanatory variable. Using the lagged version of the baseline models is quite intuitive, since the link travel time is highly related to the travel time of the predecessor links. This could be because of the fact that traffic patterns, and demand of passengers in each link are highly correlated to those of the predecessor links. However, the main purpose of using the lagged versions of the baseline models is to show that with the same amount of information the proposed model learns more complicated information. The lagged version of the two baseline models have the same structure as the simple baseline models, however in addition to the independent variables we also feed the lagged vector to the model.

5 RESULTS AND DISCUSSION

For evaluating the models performance, we use three different measures. First, is Mean Absolute Percentage Error (MAPE), which measures the error as a percentage of the true value. It is calculated as the average of the unsigned percentage error. It is calculated as

$$\text{MAPE} = \frac{100}{n} \sum_{i=1}^{n} \left| \frac{y_i - \hat{y}_i}{y_i} \right|, \tag{1}$$

where y_i is the actual travel time, and $\hat{y}_i$ is the predicted travel time for each data point. Second, we calculate the R-squared, which is the percentage of the response variable variation that is explained by a model. Its formula is given by

$$\text{R}^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}. \tag{2}$$

Finally, the Mean Squared Error (MSE), measures the average of the squares of the errors. It is calculated as

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2. \tag{3}$$

The MSE measures the quality of a predictor model, it is always positive, and values closer to zero are better. It also has the same units of measurement as the square of actual values. Since the MSE is scale-dependent, it cannot be used across different scales, however in our case, it can be a good measure for comparison between different models. The results of our experiments are shown in Tables 1 and 2 for the bus route and the streetcar, respectively.

	LR	LR-lagged	MLP	MLP-lagged	LSMT_MLP
R-squared	0.448	0.52513	0.506	0.599	**0.635**
MAPE	33.604	24.497	27.08	21.33	**21.07**
MSE	146.102	128.812	128.685	102.21	**92.846**

Table 1: Bus results

	LR	LR-lagged	MLP	MLP-lagged	LSMT_MLP
R-squared	0.343	0.367	0.517	0.509	**0.527**
MAPE	33.620	31.278	24.18	23.96	**22.00**
MSE	160.291	145.799	113.252	113.071	**110.438**

Table 2: Streetcar results

Tables 1 and 2 show that the LSTM-MLP model outperforms all other models. Note that our observed MAPE measure of the model are relatively high compared to other similar works that have been reported in this context [6, 10, 30, 31]. However, our approach is designed to predict travel time between any two consecutive stops at any time, which has a much higher variance than the travel time between a fixed pair of stops, or the total route time that other models in the literature aim at predicting. For example, [31] developed a specific ANN model for each hour of day for travel time prediction using only GPS data and reported an average MAPE of 18 %, but in their case study, the model is calibrated and tested for a specific 2-hour window of the day, whereas our considers traffic fluctuations throughout the whole day.

Table 1 reports the results for the bus dataset. In this case, the lagged version of the models are clearly superior to the simple ones. Furthermore, the combined LSMT-MLP approach is significantly better than the lagged linear regression, and it also provides a notable improvement over the lagged version of the MLP model. On the other hand, Table 2 shows that, in the case of the streetcar dataset, there is not a large difference between the simple models (LR and MLP) and their lagged counterparts. Further, the improvement due to the proposed combined approach is even more modest. We can explain this by noting that streetcars operate on a rail system with a dedicated lane, hence facing significantly less effect from traffic, and the corresponding variability. However, these results confirm that the combined LSTM-MLP approach can be effectively applied for different modes of transit and it performs at least as well as the state-of-the-art models.

Model	MLP	MLP-lagged	LSMT_MLP
time (seconds) Bus	0.021	0.044	0.098
time (seconds) Streetcar	0.035	0.065	0.128

Table 3: Model run times

Table 3 reports the running time for the three model architectures. The time values reported are the averages per epoch in seconds. As reported elsewhere in the literature (eg., [32]), due to the amount of computations in each LSTM cell and as well as the number of look-backs, training an LSTM model may be computationally expensive. However, in our case, the difference between basic MLP and LSTM-MLP, while significant, does not seem prohibitive (especially if GPU computing is utilized). Further, once fully trained and fine tuned, the predictor itself can be used in real time.

Figure 5 presents the travel time (in seconds) for prediction outputs of the MLP-lagged (orange), LR-lagged (green), and MLP-LSTM (red) models along with the ground truth (blue) for two successive cycles of a bus operating on a specific route. The graph illustrates how the MLP-LSTM model generally predicts closer values to the ground truth labels and has more predictive power compared to the other models. Indeed, one can observe that the other models smooth the predictions somewhat since they cannot fully capture the underlying dynamics of system as well as the MLP-LSTM model. At the same time, neither of the models are capable of predicting sudden spikes in the travel time in a few of the cases.

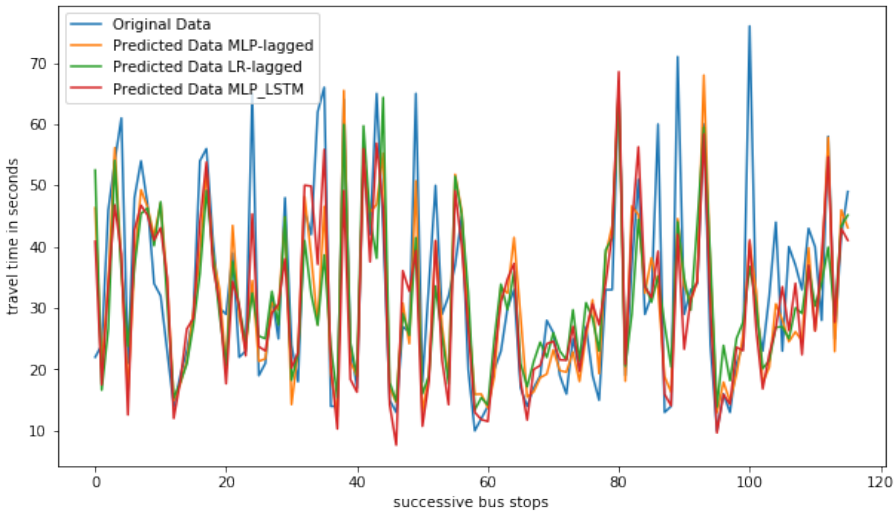


Figure 5: Travel time prediction of MLP-lagged, LR-lagged, and MLP-LSTM model versus actual data.

6 SUMMARY AND CONCLUSIONS

The problem of predicting travel times in public transit has long been recognized as important and difficult. Researchers have proposed several methodologies for such predictions, however, achieving consistently accurate predictions remains a challenge. Recent work has shown that data-driven, machine learning approaches can produce better results that traditional statistical methods such as regression or time-series analysis.

The main contribution of this work is the development of a new Deep Learning approach to predict arrival times in public transit. Our methodology is novel because it combines a multi-layer perceptron model which uses features and a long short-term memory (LSTM) neural network which uses time series data. Our model is different from those in the literature because existing approaches only focus on either feature-based prediction or time series-based prediction.

We used data from New Orleans Regional Transit Authority to calibrate and test our model. The computational study showed that our approach achieves more accurate predictions for arrival times than either a model based solely on features or time series data. Most importantly, we observe that including lagged parameters, i.e., considering time series properties in combination with the independent variables, improves prediction of bus arrival times regardless of the underlying model (linear regression or neural network). The study can be viewed then as illustrating a potential way forward to more accurate predictive models. Specifically, we already achieve some improvement by considering a deep learning framework that combines neural networks with time series analysis. A future study with more data is required to arrive at a specific predictive model architecture along with implementation details. Furthermore, the experiments show that our methodology works on different modes of transit, specifically, we tested it on both buses and streetcars, obtaining similar results. Finally, the experiments confirmed two observations noted by previous work: (1) machine learning approaches far out-perform traditional statistical models, and (2) including lagged variables improves prediction accuracy.

References

[1] G Cookson and B Pishue. Inrix global traffic scorecard. inrix research, 2018.

[2] Raghunath Nambiar, Rajesh Shroff, and Shane Handy. Smart cities: Challenges and opportunities. In *2018 10th International Conference on Communication Systems & Networks (COMSNETS)*, pages 243–250. IEEE, 2018.

[3] Jonathan English. Why public transportation works better outside the u.s., 2018.

[4] Carol L Schweiger. Real-time bus arrival information systems. *Transportation Research Board*, (48), 2003.

[5] Steven I-Jy Chien, Yuqing Ding, and Chienhung Wei. Dynamic bus arrival time prediction with artificial neural networks. *Journal of Transportation Engineering-asce - J TRANSP ENG-ASCE*, 128, 09 2002.

[6] R. Jeong and R. Rilett. Bus arrival time prediction using artificial neural network model. In *Proceedings. The 7th International IEEE Conference on Intelligent Transportation Systems (IEEE Cat. No.04TH8749)*, pages 988–993, Oct 2004.

[7] Yanjie Duan, Yisheng Lv, and Fei-Yue Wang. Travel time prediction with lstm neural network. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pages 1053–1058, Nov 2016.

[8] Xiangdong Ran, Zhiguang Shan, Yufei Fang, and Chuang Lin. An lstm-based method with attention mechanism for travel time prediction. *Sensors*, 19(4), 2019.

[9] Haitham Al-Deek, Giuseppe Pizzo, and Morgan C. Wang. Travel time prediction with non-linear time series. 1998.

[10] Jayakrishna Patnaik, Steven Chien, and Athanassios Bladikas. Estimation of bus arrival times using apc data. *Journal of public transportation*, 7(1):1, 2004.

[11] Mei Chen, Xiaobo Liu, Jingxin Xia, and Steven I-Jy Chien. A dynamic bus-arrival time prediction model based on apc data. 2004.

[12] Lelitha Vanajakshi, Shankar C Subramanian, and R Sivanandan. Travel time prediction under heterogeneous traffic conditions using global positioning system data from buses. *IET intelligent transport systems*, 3(1):1–9, 2009.

[13] Vivek Kumar, B Anil Kumar, Lelitha Vanajakshi, and Shankar C Subramanian. Comparison of model based and machine learning approaches for bus arrival time prediction. In *Proceedings of the 93rd Annual Meeting*, pages 14–2518. Transportation Research Board, 2014.

[14] Wei Fan and Zegeye Gurmu. Dynamic travel time prediction models for buses using only gps data. *International Journal of Transportation Science and Technology*, 4(4):353–366, 2015.

[15] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997.

[16] C. L. Giles, G. M. Kuhn, and R. J. Williams. Dynamic recurrent neural networks: Theory and applications. *IEEE Transactions on Neural Networks*, 5(2):153–156, March 1994.

[17] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3156–3164, 2014.

[18] Hasim Sak, Andrew W. Senior, and Françoise Beaufays. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In *INTERSPEECH*, 2014.

[19] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3104–3112. Curran Associates, Inc., 2014.

[20] Jintao Ke, Hongyu Zheng, Hai Yang, and Xiqun Michael Chen. Short-term forecasting of passenger demand under on-demand ride services: A spatio-temporal deep learning approach. *Transportation Research Part C: Emerging Technologies*, 85:591–608, 2017.

[21] Tara N Sainath, Oriol Vinyals, Andrew Senior, and Haşim Sak. Convolutional, long short-term memory, fully connected deep neural networks. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4580–4584. IEEE, 2015.

[22] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[23] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *ICML*, 2012.

[24] John J Bartholdi III and Donald D Eisenstein. A self-coordinating bus route to resist bus bunching. *Transportation Research Part B: Methodological*, 46(4):481–491, 2012.

[25] Yun Ye and Jie Li. Analyzing data from avl/apc system for improving transit management - theory and practice. volume 2, pages 267–274, 01 2011.

[26] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[27] Martin Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. Tensorflow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, pages 265–283, 2016.

[28] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[29] François Chollet et al. Keras. `https://keras.io`, 2015.

[30] J. Liu, W. Wang, X. Gong, X. Que, and H. Yang. A hybrid model based on kalman filter and neutral network for traffic prediction. In *2012 IEEE 2nd International Conference on Cloud Computing and Intelligence Systems*, volume 02, pages 533–536, Oct 2012.

[31] Zegeye Kebede Gurmu and Wei Fan. Artificial neural network travel time prediction model for buses using only gps data. *Journal of Public Transportation*, 17:45–65, 06 2014.

[32] Haşim Sak, Andrew Senior, and Françoise Beaufays. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In *Fifteenth annual conference of the international speech communication association*, 2014.